ALGORITMOS GENÉTICOS APLICADOS A LA CATEGORIZACIÓN AUTOMÁTICA DE DOCUMENTOS

Yolis, E.[1], Britos, P. [2,3], Sicre, J.[2], Servetto, A. [4,3], García-Martínez, R.[2,1] y Perichinsky. G.[4,3]

1.- Laboratorio de Sistemas Inteligentes
Facultad de Ingeniería.
Universidad de Buenos Aires.
Paseo Colón 850 4to Piso. Ala Sur. (1063) Capital Federal

2.- Centro de Ingeniería del Software e Ingeniería del Conocimiento (CAPIS)
Instituto Tecnológico de Buenos Aires
Av. Madero 399. (1106) Capital Federal.

3.- Programa de Doctorado en Ciencias Informáticas
Facultad de Informática
Universidad Nacional de La Plata.
Buenos Aires

4.- Laboratorio de Sistemas Operativos y Bases de Datos.
Facultad de Ingeniería.
Universidad de Buenos Aires
Paseo Colón 850 4to Piso. Ala Sur. (1063) Capital Federal

Resumen - La categorización automática de documentos ha estado recibiendo creciente atención debido al incremento en la cantidad de información disponible en forma electrónica y a la necesidad cada vez mayor de encontrar la información buscada en un tiempo mínimo. Si bien existen numerosos algoritmos para categorizar documentos, todos ellos evaluan un subconjunto pequeño del espacio de posibles soluciones. Este trabajo presenta un algoritmo genético adaptado al problema de categorización de documentos. El algoritmo propuesto introduce 5 nuevos operadores, diseñados específicamente para la resolución del problema de categorización. Los resultados obtenidos demuestran que el algoritmo genético logra explorar el espacio de búsqueda más amplia y eficientemente que los algoritmos previos tomados como referencia.

Palabras clave: Categorización Automática de Documentos, Algoritmos Genéticos, Computación Evolutiva

1 Introducción

La *Categorización de Documentos* (Document Clustering) puede definirse como la tarea de separar documentos en grupos. El criterio de agrupamiento se basa en las similitudes existentes entre ellos [Kaufmann *et al.*, 1990].

La categorización automática de documentos se comenzó a investigar dentro de la rama de "Recuperación de Información" (en inglés , "Information Retrieval*"*). En el contexto del recupero de información, Van Rijsbergen formula en 1979 la denominada "Hipótesis del Agrupamiento" (en inglés, "Cluster Hypothesis*"*) [Van Rijsbergen, 1979]. Básicamente, la hipótesis del agrupamiento sostiene que "los documentos fuertemente asociados tienden a ser relevantes para la mísma consulta". Basándose en esta hipótesis, la categorización automática de documentos tiene en cuenta el contenido de los documentos para agruparlos, ya que documentos similares contendrán palabras (términos) similares.

Sus aplicaciones más importantes son:

- Mejorar el rendimiento de los motores de búsqueda de información mediante la categorización previa de todos los documentos disponibles [Van Rijsbergen, 1979; Dunlop y Van Rijsbergen, 1991; Faloutsos y Oard, 1995].

- Facilitar la revisión de resultados por parte del usuario final, agrupando los resultados luego de realizar la búsqueda [Croft, 1978; Cutting *et al.*, 1992; Allen *et al.*, 1993; Leousky y Croft, 1996].

2 Estado del arte

Una definición del problema del agrupamiento de documentos (que es aplicable al agrupamiento de cualquier tipo de elementos), se puede enunciar de la siguiente manera (adaptada de [Zhao *et.al.*, 2001]): Dado un conjunto *S*, de *N* documentos, se quiere encontrar la partición $S_1, S_2, ..., S_k$, tal que cada uno de los *N* documentos se encuentre sólamente en un grupo S_i, y que cada documento sea más similar a los documentos del mismo grupo que a los documentos asignados a los otros grupos.

Para poder definir medidas de semejanza entre los objetos a agrupar, éstos se representan mediante vectores $v = (a_1 , a_2 , ... , a_m)$, donde cada componente del vector es el valor de un atributo del objeto. De esta forma, cada uno de los objetos a agrupar es un punto en un Espacio Euclideano de *m* dimensiones, R^m.

Se define al centroide C_s de un grupo de elementos *S*, que contiene *h* elementos s_i como [Steinbach *et.al.*, 2000; Zhao *et.al.*, 2001]: $C_s = \frac{\sum_{i=1}^{h} s_i}{h}$, lo que es simplemente el promedio de los vectores que componen el grupo.

En el modelo de representación vectorial, siendo $\|v\|$ la longitud (norma) del vector *v*, una de las medidas de semejanza más utilizadas [Cole, 1998; Steinbach *et.al.*, 2000; Strehl *et.al.*, 2000; Zhao *et.al.*, 2001] es el Coeficiente del coseno extendido: $\cos(d_1, d_2) = \frac{d_1 \bullet d_2}{\|d_1\| * \|d_2\|}$. Esta medida tiene la propiedad de no depender del tamaño de los documentos, ya que $\mathbf{cos(d_1,d_2) = cos(a.d_1,d_2)}$ para a>0. Sin embargo, los documentos se normalizan para que tengan longitud unitaria, ya que entonces, $\cos(d_1, d_2) = d_1 \bullet d_2$, y la semejanza entre los documentos se puede calcular como el producto vectorial entre ellos.

2.1 Métodos para categorizar documentos

Las formas de clasificación de objetos, tales como asignar clases predeterminadas a cada elemento ó agruparlos en forma significativa, son susceptibles de dividirse según el esquema de la figura 1:

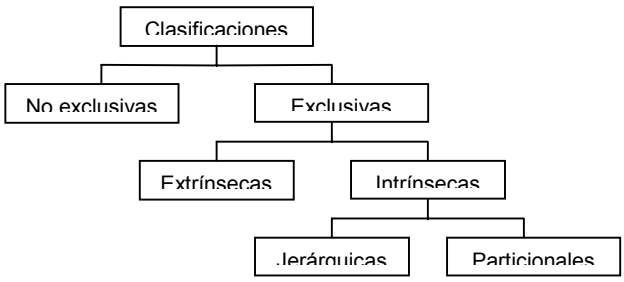


Figura 1 - División de las formas de clasificar objetos

- **No exclusivas** : Un mismo objeto puede pertenecer a varios grupos.
- **Exclusivas** : Cada objeto pertenece solamente a un grupo.

- **Extrínsecas (supervisadas)** : Las clases a las que pertenecen los objetos están predefinidas, y se conocen ejemplos de cada una, ó algunos de los objetos ya están clasificados y son utilizados por el algoritmo para aprender a clasificar a los demás.
- **Intrínsecas (no supervisadas)** : La clasificación se realiza en base a las características propias de los objetos, sin conocimiento previo sobre las clases a las que pertenecen.

La categorización automática de documentos se encuentra en la categoría “intrínseca”, ya que los criterios de agrupamiento se basan en la información contenida en los mismos para determinar sus similitudes

2.1.1 Métodos Jerárquicos

Los algoritmos jerárquicos se caracterizan por generar una estructura de árbol (llamada "dendograma"), en la que cada nivel es un agrupamiento posible de los objetos de la colección [Jain *et.al.*, 1999; Dash *et.al.*, 2001; Han, 2001]. Cada vértice (nodo) del árbol es un grupo de elementos. La raiz del árbol (primer nivel) se compone de un sólo grupo que contiene todos los elementos. Cada hoja del último nivel del árbol es un grupo compuesto por un sólo elemento (hay tantas hojas como objetos tenga la colección). En los niveles intermedios, cada nodo del nivel n es dividido para formar sus hijos del nivel $n + 1$.

Si bien se acepta que los métodos jerárquicos aglomerativos obtienen resultados de buena calidad [Cutting *et.al.*, 1992; Cole, 1998; Dash *et.al.*, 2001], los tiempos de cómputo que requieren son inaceptables para la mayoría de las aplicaciones.

2.1.2 Métodos particionales

Los métodos de optimización o particionales, a diferencia de los jerárquicos, no van generando distintos niveles de agrupamiento de los objetos, sino que trabajan en un sólo nivel, en el que se refina (optimiza), un agrupamiento [Everitt, 1993]. Estos métodos asumen que el valor de k (la cantidad de grupos), está definida de antemano [Qin He, 1996].

Estos algoritmos son muy similares entre sí, por lo que se describirá únicamente el algoritmo "k-means", que, además de ser el referente más típico, en la bibliografía es el que más frecuentemente se encuentra aplicado al campo de categorización automática de documentos [Steinbach *et.al.*, 2000].

2.1.2.1 Algoritmo "K-means"

Este algoritmo, presentado originalmente por [McQueen, 1967], utiliza a los centroides de cada grupo como sus puntos representantes.

Partiendo de una selección inicial de k centroides, cada uno de los elementos de la colección se asigna al grupo con el centroide más cercano. A continuación, se calcula el centroide de cada uno de los grupos resultantes. En los primeros pasos se obtienen las mayores diferencias entre los centroides originales y los calculados luego de las reasignaciones. Los puntos de la colección vuelven a asignarse al grupo del centroide más cercano, y estos pasos se repiten hasta que los k centroides no cambian luego de una iteración (ésto es equivalente a decir que el valor de la función utilizada como criterio de optimización no varía).

El algoritmo K-Means es mucho más eficiente que los métodos jerárquicos (los tiempos de cómputo requeridos son lineales con la cantidad documentos a agrupar [Han *et.al.*, 2001]), pero es dependiente de la selección inicial de centroides [Bradley, 1998]. Sus resultados pueden ser bastante pobres y suelen variar mucho si se aplica varias veces a la misma colección de documentos, ya que si la selección de centroides al azar es mala, la solución encontrada también lo será.

2.2 El algoritmo "Bisecting K-Means"

Varios investigadores han presentado alternativas a los algoritmos tradicionales de categorización automática de documentos [Cutting *et.al.*, 1992; Bradley *et.al.*, 1998; Steinbach *et.al.*, 2000]. El algoritmo "Bisecting K-Means" es una de ellas [Steinbach *et.al.*, 2000]. En este trabajo se decidió tomar esta variante como referencia para compararla con la solución propuesta, por varios motivos:

- El trabajo que la presenta es uno de los más recientes, (año 2000), y esta variante se analiza tambien en un trabajo del año 2001 [Zhao *et.al.*, 2001].

- Uno de los autores (que participa en ambos trabajos) es una personalidad reconocida dentro del dominio del recupero de información y de la categorización automática, con más de 70 trabajos publicados, y varios desarrollos de software relativos al tema.

- Los trabajos demuestran que la calidad de los resultados supera tanto a los algoritmos jerárquicos aglomerativos como al "K-Means", con tiempos de cómputo lineales con la cantidad de elementos a agrupar.

El algoritmo "Bisecting K-Means" sigue los siguientes pasos:

1) Colocar todos los documentos en un sólo grupo
2) Elegir el grupo más grande para dividirlo.
3) Dividir el grupo en 2 subgrupos usando el algoritmo "K-Means" (paso de bisección).
4) Repetir el paso de bisección 5 veces, y tomar la división que lleve a una mayor similitud promedio.
5) Repetir los pasos 2, 3 y 4 hasta alcanzar el número de grupo deseado.

2.2.1 Algoritmo "Bisecting K-Means con refinamiento"

Esta variante consiste en refinar la solución obtenida con el algoritmo "Bisecting K-Means" utilizando el algoritmo "K-Means" estándar. El agrupamiento generado por el algoritmo "Bisecting K-Means" se toma como el punto de partida para el algoritmo "K-Means". Este refinamiento apunta a corregir posibles asignaciones incorrectas de documentos a los grupos que se podrían haber realizado durante las fases de división del algoritmo. Los resultados muestran que el paso de refinamiento mejora la calidad de las soluciones obtenidas [Steinbach *et.al.*, 2000].

Aunque el algoritmo "Bisecting K-Means" ha demostrado obtener mejores soluciones que el algoritmo k-means y los métodos jerárquicos aglomerativos [Steinbach *et.al.*, 2000], en tiempos de cómputo lineales con la cantidad de documentos a agrupar, la forma en que explora el espacio de búsqueda es claramente sub-óptima. Al momento de dividir un grupo, el algoritmo "Bisecting K-Means" realiza 5 divisiones diferentes del grupo, y luego elige una de ellas, descartando las demás divisiones que creó. El siguiente análisis muestra por qué esta forma de trabajo no es eficiente:

- Cuando el grupo es fácilmente divisible (hay 2 subgrupos claramente definidos), las 5 corridas del algoritmo K-Means encontrarán la misma división, o divisiones con muy pocas variaciones. En este caso, el algoritmo estaría repitiendo 5 veces el mismo trabajo.
- Cuando no hay dos subgrupos claramente definidos, las 5 corridas del algoritmo K-Means encontrarán divisiones diferentes. Es muy posible que en alguna de ellas haya encontrado un grupo excelente y uno muy malo, que en promedio, hagan que descarte esa división, y no tiene ningún mecanismo que le permita aprovechar el trabajo realizado, tomando alguna característica positiva de esa división.

2.3 Introducción a los Algoritmos Genéticos

Los algoritmos genéticos fueron desarrollados por John Holland, junto a su equipo de investigación, en la universidad de Michigan en la década de 1970 [Holland, 1975].

Los algoritmos genéticos combinan las nociones de supervivencia del más apto con un intercambio estructurado y aleatorio de características entre individuos de una población de posibles soluciones, conformando un algoritmo de búsqueda que puede aplicarse para resolver problemas de optimización en diversos campos [Goldberg, 1989].

Imitando la mecánica de la evolución biológica en la naturaleza, los algoritmos

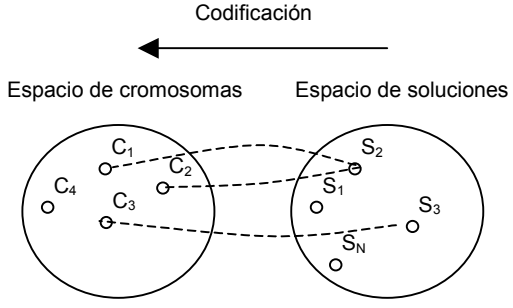


Figura 2 - Cada cromosoma codifica una posible solución al problema

genéticos operan sobre una población compuesta de posibles soluciones al problema. Cada elemento de la población se denomina “cromosoma”. Un cromosoma es el representante, dentro del algoritmo genético, de una posible solución al problema. La forma en que los cromosomas codifican a la solución se denomina “Representación”.

La figura 3 representa el esquema de funcionamiento del algoritmo genético.

El proceso comienza seleccionando un número de cromosomas para que conformen la población inicial.

A continuación se evalúa la función de adaptación para estos individuos. La función de adaptación da una medida de la aptitud del cromosoma para sobrevivir en su entorno. Debe estar definida de tal forma que los cromosomas que representen mejores soluciones tengan valores más altos de adaptación.

Los individuos más aptos se seleccionan en parejas para reproducirse. La reproducción genera nuevos cromosomas que combinan características de ambos padres. Estos nuevos cromosomas reemplazan a los individuos con menores valores de adaptación.

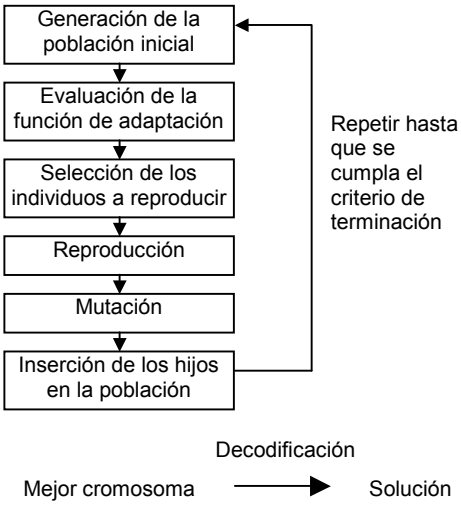


Figura 3 - Esquema de funcionamiento del algoritmo genético

A continuación, algunos cromosomas son seleccionados al azar para ser mutados. La mutación consiste en aplicar un cambio aleatorio en su estructura.

El ciclo de selección, reproducción y mutación se repite hasta que se cumple el criterio de terminación del algoritmo, momento en el cual el cromosoma mejor adaptado se devuelve como solución.

3 Utilización de algoritmos genéticos para la categorización automática

La aplicación de algoritmos genéticos al problema general de la categorización automática se ha investigado con interés [Jones *et.al.*, 1991; Bezdek *et.al.*, 1994; Estivill-Castro *et.al.*, 1997; Cole, 1998; Jain *et.al.*, 1999; Hall *et.al.*, 1999; Falkenauer, 1999; Painho *et.al.*, 2000]. Sin embargo, son muy pocos los autores que los han aplicado a la categorización automática de documentos [Jones *et.al.*, 1995]. Esto se debe a que, pese a que los algoritmos genéticos han demostrado capacidad de explorar el espacio de búsqueda amplia y eficientemente [Goldberg, 1989; Koza, 1997; Falkenauer, 1999], los resultados de las investigaciones no han dado buenos resultados al aplicarlos a la categorización automática en general, ni a la categorización automática de documentos en particular. La causa de estos fracasos radica en la aplicación inadecuada de los algoritmos genéticos al problema. Se ha hecho notar [Falkenauer, 1999; Estivill-Castro, 2000] que los algoritmos genéticos no son una solución universal que pueda aplicarse en forma pura a cualquier problema, sino un paradigma que debe adaptarse a los problemas concretos a los que se aplica.

Otro punto importante, ignorado por la mayoría de los autores, es la posibilidad de incorporar en los operadores del algoritmo genético conocimiento específico del problema al cual se está aplicando el mismo. De esta manera, los operadores no trabajan en forma totalmente aleatoria sobre los individuos de la población, sino que tienden a realizar cambios que apunten a buenas soluciones para el problema a resolver. Sin la aplicación de conocimiento específico del problema

que se está resolviendo en los operadores, es casi imposible que el algoritmo genético supere a los algoritmos diseñados *ad hoc* para el problema [Goldberg, 1989; Estivill-Castro, 2000].

4 Algoritmo propuesto

En esta sección se describe el algoritmo génetico de categorización automática de documentos propuesto. Este algoritmo presenta un nuevo operador de cruza y cuatro nuevos operadores de mutación diseñados específicamente para el problema de la categorización automática de documentos.

4.1 Representación

La representación elegida fue la Numeración de grupo. Este método es el más frecuentemente utilizado en la literatura [Falkenauer, 1999].

La estructura del cromosoma utilizado en el algoritmo propuesto es la siguiente:

- **Vector de elementos (1..N)** : es el vector que contiene, para cada elemento, el número de grupo al que pertenece
- **CantGrupos** : es la cantidad de grupos que tiene el agrupamiento
- **Vector de grupos (1..CantGrupos)**
  - **CantElementos** : es la cantidad de elementos que tiene el grupo
  - **NroGrupo** : es el número que identifica al grupo
  - **VecCentroide** : es el centroide del grupo

4.2 Generacion de la población inicial

Para seleccionar el método de generación de la población inicial se hicieron diversas pruebas y se extrajeron las siguientes observaciones:

- El tamaño de la población requerido para que el algoritmo genético encuentre buenas soluciones es más chico cuanto mejores sean los agrupamientos de la población inicial.
- Generar la población inicial al azar es lo más rápido, pero los agrupamientos generados son de muy baja calidad, y lleva demasiadas generaciones al algoritmo genético llegar a buenas soluciones.
- Utilizando el algoritmo k-means (el más rápido de los algoritmos conocidos que dan soluciones aceptables) se obtienen buenas soluciones en relativamente pocas generaciones, pero el proceso de generación de la población inicial exige demasiado tiempo (aún teniendo en cuenta que el tamaño de la población inicial puede reducirse), ya que el algoritmo k-means es del orden de (n.k).

Buscando una solución intermedia, se decidió aplicar el algoritmo k-means para generar los agrupamientos de la población inicial, pero en lugar de formar k grupos, se generan agrupamientos de $\log(k)$ grupos. De esta forma, la generación de la población inicial lleva tiempos del orden de $(n.\log(k))$. El algoritmo genético tiene operadores que se ocupan de ir dividiendo a los grupos para llegar a soluciones que tengan k grupos, que se detallan más adelante.

4.3 Funcion de adaptación

Para el algoritmo propuesto se eligió la Similitud promedio del agrupamiento como función de adaptación. Esta función se detalla en la sección 5 (“Prueba Experimental”).

4.4 Seleccion

Para el algoritmo propuesto se utiliza el método de Torneo, por los siguientes motivos:

- Es independiente de la escala de la función de adaptación
- No requiere ordenar los valores de adaptación de cada cromosoma
- Permite regular la presión de selección

4.5 Cruza

Dado que ninguno de los operadores de cruza encontrados en la literatura daba solución a las cuestiones planteadas en la sección 3, se diseñó un nuevo operador de cruza que cumpliera con las características buscadas.

4.5.1 Cruza Pasa Grupo

El operador diseñado se denomina “Cruza Pasa Grupo”. El operador genera 2 hijos a partir de los padres. Primero asigna un rol a cada padre para generar el primer hijo, y luego invierte los roles para generar el otro.

El agrupamiento de uno de los padres es copiado al hijo sin modificaciones, y luego uno de los grupos del segundo padre se pasa al hijo, reacomodando los elementos que hagan falta.

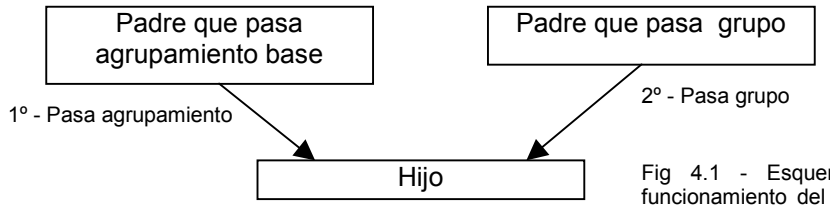


Fig 4.1 - Esquema básico del funcionamiento del operador Cruza Pasa Grupo

Los pasos que sigue el algoritmo para generar uno de los hijos son:

1. Crear un cromosoma con el mismo agrupamiento que el primer padre (el padre que pasa el agrupamiento)
2. Seleccionar un grupo fuente del segundo padre (el padre que pasa el grupo)
3. Seleccionar un grupo destino en el hijo, que va a transformarse en el grupo del padre
4. Reacomodar los elementos del hijo según sea necesario para que el grupo destino contenga los elementos que contiene el grupo fuente

4.6 Mutación

Durante las pruebas realizadas se observó que los operadores de mutación clásicos retrasaban considerablemente la convergencia del algoritmo genético, ya que lo obligaban a explorar regiones del espacio de búsqueda al azar, donde la mayor parte de las veces no se encontraban mejores soluciones. Al diseñar un algoritmo genético que pueda competir en términos de eficiencia con los algoritmos específicos del dominio, debe sacrificarse en parte la amplitud de la exploración, restringiéndola a aquellas regiones donde es más probable que haya buenas soluciones. Esto puede lograrse diseñando operadores de mutación que no hagan cambios totalmente aleatorios, sino que, de la misma forma que el operador de cruza, tiendan a hacer cambios que aumenten la calidad de las soluciones, dejando algunas pocas decisiones libradas al azar.

A continuación se describen los 4 nuevos operadores de mutación diseñados para el algoritmo propuesto.

4.6.1 Mutación RefinarKM

Cuando se muta un cromosoma mediante este operador, se aplica al agrupamiento lo que se denomina "Refinamiento ligero", que es una modificación del algoritmo k-means.

El refinamiento ligero consiste en una sóla iteración del algoritmo k-means. Pero para cada elemento, en lugar de buscar entre los *k* grupos del agrupamiento aquel cuyo centroide sea más cercano al del elemento, la búsqueda se realiza entre 2**log(k)* grupos seleccionados al azar. De esta forma, en lugar de ser del orden de (n.k), la aplicación del operador tiene una complejidad del orden de (n.2.log(k)).

4.6.2 Mutación Refinar Selectivo

Al aplicar este operador a un cromosoma, se intenta mejorar la calidad de los mejores grupos del agrupamiento. Lo que se busca es que el agrupamiento mutado tenga algunos grupos de muy buena calidad que luego, si el cromosoma es seleccionado como padre, pasen al hijo. El operador identifica los grupos con mayor y menor similitud promedio y luego recorre los mejores grupos en busca de los elementos que están más alejados del centroide del grupo. Esos elementos son removidos del grupo (lo que aumenta su similitud promedio), y son reasignados a alguno de los grupos con menor similitud promedio (lo que disminuye aún más su similitud promedio).

4.6.3 Mutación Split

Como en la generación de la población inicial los agrupamientos se generan con log(k) grupos, este operador es necesario para que los agrupamientos lleguen a tener *k* grupos. El operador selecciona un grupo y lo divide en 2. Si al cromosoma se le aplicó anteriormente el operador de mutación Join (que se describe más adelante), el grupo formado por este operador es el que se divide. En caso contrario, el grupo a dividir es el que tiene mayor tamaño. Una vez que se tiene el grupo a dividir, se lo divide en 2 aplicando a los elementos del grupo la primera iteración (fase inicial) del algoritmo k-means, con k = 2.

4.6.4 Mutación Join

La aplicación de este operador a un cromosoma que tenga *k* grupos hará que en la iteración siguiente se aplique el operador de mutación Split (ya que el agrupamiento tendrá *k-1* grupos). El operador apunta a juntar en un solo grupo dos grupos de mediana o mala calidad, con la esperanza de que el operador de mutación Split pueda armar grupos de mejor calidad, mejorando la solución. El operador elige dos grupos del agrupamiento, y junta en un sólo grupo los elementos de ambos.

4.8 Tamaño de la población

Los tamaños de población utilizados en trabajos previos varían entre valores de 40 a 1000 individuos [Cole, 1998]. Se ha hecho notar [Estivill-Castro, 2000] que tamaños de población excesivamente grandes retrasan la convergencia del algoritmo genético, impidiéndole competir con otro tipo de métodos. Esto fue confirmado por las pruebas realizadas.

En el algoritmo propuesto, se comienza con una población inicial de 12 individuos, que luego disminuye a 10 cuando se han completado el 40% de las generaciones previstas.

4.9 Criterio de terminación

En las pruebas realizadas se observó que la calidad de la mejor solución aumentaba con cierto ritmo hasta un punto, para luego crecer muy lentamente hasta alcanzar la convergencia. Para evitar esa espera, se decidió terminar la ejecución del algoritmo genético luego de una cantidad fija de iteraciones. Experimentalmente se encontró el valor GEN_MAX = (N / 20) + 25, donde N es la cantidad de documentos a categorizar.

4.10 Algoritmo Genetico con Refinamiento

Al igual que en el caso del algoritmo “Bisecting K-Means con refinamiento”, esta variante consiste en refinar la solución obtenida utilizando el algoritmo “K-Means” estándar.

5 Prueba experimental

Para la realización de los experimentos, se utilizaron datos de la Colección de Prueba para Categorización de Documentos “Reuters 21578” [Lewis, 1997]. Ésta colección se ha convertido en un estándar “de facto” dentro del dominio de la categorización automática de documentos y es utilizada por numerosos autores de la materia [Joachims, 1998; Yang *et.al.*, 2000; Steinbach *et.al.*, 2000; Zhao *et.al.*, 2001].

5.1 Medidas de evaluación

A continuación se detallarán las diversas medidas que se utilizarán para evaluar los agrupamientos.

5.1.1 Similitud promedio

La similitud promedio de un grupo es la que surge de promediar la similitud existente entre todos los documentos del grupo tomados de a pares. Por ejemplo, para el grupo j, que tiene n_j elementos, utilizando el coeficiente del coseno extendido como medida de similitud, $sim\_prom_j = \frac{1}{n_j^2}\left(\sum_{\substack{i=1\\ j=1}}^{n_j} \frac{d_i \bullet d_j}{\|d_i\| * \|d_j\|}\right)$. La similitud promedio de todo el agrupamiento se obtiene sumando la similitud promedio de cada grupo multiplicada por la cantidad de elementos del grupo, y dividiendo el total por la cantidad total de elementos de la muestra, $sim\_prom = \frac{\sum_{j=i}^{k} n_j * sim\_prom_j}{N}$

Esta medida da cuenta de cuánto tienen en común los elementos de cada grupo. Cuanto más homogeneo sea cada grupo, mayor valor tendrá este parámetro. Valores más grandes de similitud promedio indican una mejor calidad del agrupamiento.

5.1.2 Entropía

El concepto de entropía tiene su origen en el campo de la física ,y es utilizada como medida del grado de desorden de un sistema [Carter, 2000]. En 1948, Shannon [Shannon, 1948] incorpora el término a la teoría de la información como una función que mide la cantidad de información generada por un proceso.

Para medir la calidad de un agrupamiento utilizando la entropía, primero se calcula entropía de cada grupo. Para cada categoría “real” i, se calcula la probabilidad p_{ij} de que un miembro del grupo j sea de la categoría i, $p_{ij} = n_{ij} / n_j$, donde n_{ij} es la cantidad de documentos de la categoría i que se encuentran en el grupo j y n_j es la cantidad de documentos del grupo j.

La entropía de cada grupo se calcula usando la fórmula: $H_j = -\sum_i p_{ij} * \log(p_{ij})$, donde i recorre todas las categorías “reales” de los documentos.

La entropía total del agrupamiento se calcula luego ponderando la entropía de cada grupo de acuerdo a su tamaño, $H = \frac{\sum_{j=1}^{k} n_j * H_j}{N}$. La entropía tendrá un valor máximo cuando los documentos de

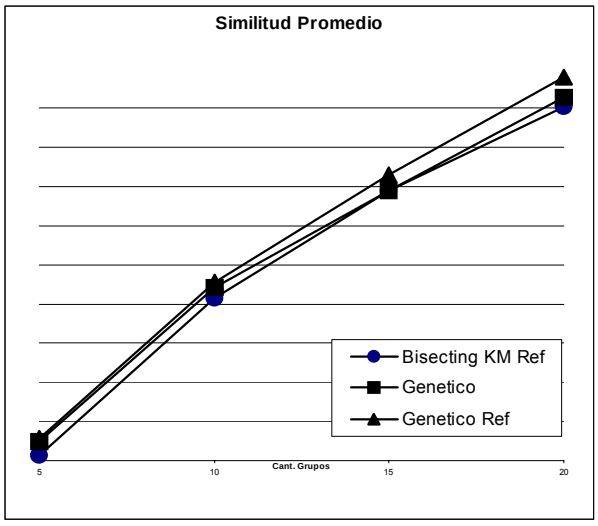
cada categoría estén distribuidos uniformemente entre los grupos, y un valor igual a 0 cuando cada categoría tenga sus documentos en un sólo grupo. Valores más chicos de entropía indican una mejor calidad del agrupamiento.

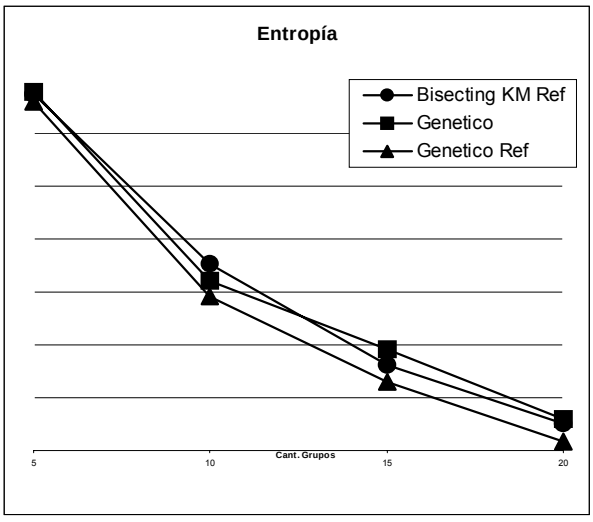
5.2.2.3 Cantidad de operaciones

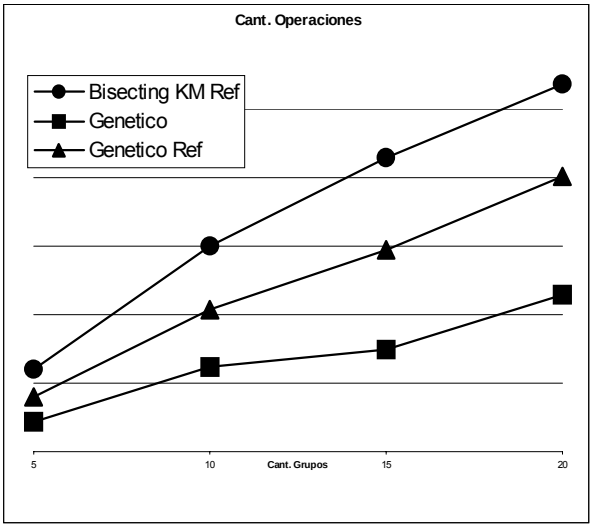
Esta medida no se relaciona con la calidad del agrupamiento obtenido, sino con la eficiencia del algoritmo. La cantidad de operaciones da una medida de la complejidad computacional del algoritmo [Wilf, 1986], ya que mide el esfuerzo computacional que llevó al algoritmo obtener el agrupamiento, en una escala que es independiente de los recursos de hardware sobre los cuales se realizaron las pruebas (velocidad del procesador, memoria, velocidad del disco, etc).

Las tres operaciones básicas que se repiten en los algoritmos evaluados son: Multiplicación de 2 vectores, Cálculo de la norma de un vector y Sumas de vectores. Para simplificar las mediciones, se supuso que las 3 operaciones eran equivalentes (multiplicar dos vectores lleva el mismo esfuerzo computacional que sumar dos vectores o calcular la norma de un vector). Por la naturaleza de las operaciones (todas iteran sobre los elementos de un vector realizando operaciones algebraicas), es una aproximación razonable. Lógicamente, un menor número de operaciones realizadas (para resultados similares) indica una mayor eficiencia del algoritmo.

5.2 Resultados







Se realizaron experimentos formando 5, 10, 15 y 20 grupos con cada uno de los algoritmos sobre subconjuntos de documentos extraidos de la colección “Reuters 21578”. Para cada medida de evaluación, se graficó la evolución de la misma en función de la cantidad de grupos para cada uno de los algoritmos a comparar: “Bisecting K-Means con refinamiento”, “Genético” y “Genético con refinamiento”.

Puede observarse en los gráficos que el algoritmo “Genético” obtiene resultados levemente superiores que el algoritmo “Bisecting K-Means” en cuanto a similitud promedio y entropía, requiriendo para ello aproximadamente un 50% menos de operaciones.

El algoritmo “Genético con refinamiento”, por su parte, supera claramente al algoritmo “Bisecting K-Means con refinamiento” de acuerdo a las medidas de similitud promedio y entropía, realizando aproximadamente un 20% menos de operaciones.

6 Conclusiones

Este trabajo propone una adaptación de un algoritmo genético al problema de la categorización automática de documentos.

La investigación y los resultados obtenidos confirman que los algoritmos genéticos son una poderosa herramienta para la resolución de problemas en los cuales el espacio de soluciones es amplio y la función de optimización es compleja.

Se ha encontrado tambien que, tal como lo han afirmado otros autores [Falkenauer, 1999; Estivill-Castro, 2000], los algoritmos genéticos no son un método de solución universal de problemas, sino un paradigma que debe adaptarse correctamente al problema a resolver. El algoritmo propuesto logra resultados efectivos porque en el diseño del mismo se han adaptado los conceptos que aplican los algoritmos genéticos y se han creado nuevos operadores específicos para el problema a resolver.

7 Bibliografía

Allen, R. B., Obry, P. y Littman, M. (1993). *An interface for navigating clustered document sets returned by queries*, Proceedings of the ACM Conference on Organizational Computing Systems.

Bezdeck, J. C., Boggavaparu, S., Hall, L. O. y Bensaid, A. (1994). *Genetic algorithm guided clustering*, in Proc. of the First IEEE Conference on Evolutionary Computation, 3440.

Bradley, P.S. y Fayyad, U.M. (1998). *Refining initial points for k-means clustering*. Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98), Morgan Kaufmann

Carter, T. (2000). *An introduction to information theory and entropy*. Complex Systems Summer School.

Cole, Rowena M. (1998). *Clustering with Genetic Algorithms*. Thesis for the degree of Master of Science, University of Western Australia.

Croft, W. B. (1978). *Organizing and searching large files of documents*, Ph.D. Thesis, Univ of Cambridge.

Cutting, D. R., Karger, D. R., Pedersen, J. O. y Tukey, J. W. (1992). *Scatter/Gather: A cluster-based approach to browsing large document collections*, Proceedings of the 15th International ACM SIGIR Conference, Páginas 318-29.

Dash, M., y Liu, H. (2001). *Efficient Hierarchical Clustering Algorithms Using Partially Overlapping Partitions*. Pacific-Asia Conference on Knowledge Discovery and Data Mining, páginas 495-506.

Dunlop, M.D. y Van Rijsbergen, C. J. (1991). *Hypermedia and free text retrieval*. RIA091 Conference, Barcelona.

Estivill-Castro, V. (2000). *Hybrid Genetic Algorithms Are Better for Spatial Clustering.* Pacific Rim International Conference on Artificial Intelligence.

Estivill-Castro, V. y Murray, A. (1997). *Spatial Clustering for Data Mining with Genetic Algorithms.* FIT, Technical Report, 97-10.

Everitt, Brian S. (1993). *Cluster analysis.* Halsted Press, 3ra edición.

Falkenauer, Emanuel. (1999). *Evolutionary Algorithms: Applying Genetic Algorithms to Real-World Problems*. Springer, New York, Pag 65--88.

Faloutsos, Christos y Oard, Douglas W. (1995). *A survey of Information Retrieval and Filtering Methods*. CS-TR3514, Univ. of Maryland.

Goldberg, David E. (1989). *Genetic Algorithms - in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Inc.
Hall, L.O., Ozyurt, B. y Bezdek, J.C. (1999). *Clustering with a genetically optimized approach*. IEEE Trans. On Evolutionary Computation.
Han, J., Kamber, M. y Tung, A.K.H. (2001). *Spatial clustering methods in data mining: A survey*. H. Miller and J. Han, editors, Taylor and Francis.
Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: The University of Michigan Press.
ISO/IEC 2382-1:1993 *Information technology -- Vocabulary* --. Part 1: Fundamental terms
Jain, A. K., Murty, M.N., y Flinn, P.J. (1999). *Data Clustering: A review*. ACM Computing Surveys, Vol. 31, Nro 3, Septiembre 1999.
Joachims, T. (1998). *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*. Proceedings of ECML-98, Heidelberg, DE.
Jones, D. R. y Beltramo, M. A. (1991) *Solving partitioning problems with genetic algorithms*. Proceedings of the fourth International Conference on Genetic Algorithms, pages 442-449.
Jones, G., Robertson, A.M., Santimetvirul, C. y Willet, P. (1995). *Non-hierarchical document clustering using a genetic algorithm*. Information Research, an electronic journal, Vol 1, No 1, April, 1995.
Kaufmann, L. y Rousseeuw, P. J. (1990). *Finding Groups in data: An introduction to Cluster Analysis*, John Wiley & Sons, Inc., NY.
Koza, John R. (1997). *Genetic Programming*. Cambridge : M.I.T. Press.
Leousky, A. V. y Croft, W. B. (1996). *An evaluation of techniques for clustering search results*, Technical Report IR-76, University of Massachusetts, Amherst.
Lewis, D. (1997). *Reuters-21578 text categorization test collection*, www.daviddlewis.com/resources/.
Maarek, Yoelle S., Fagin, Ronald, Ben-Shaul, Israel Z. y Pelleg, Dan. (2000). *Ephemeral Document Clustering for Web Applications*. IBM RJ 10186.
McQueen, J. (1967). *Some methods for classification and analysis of multivariate observations*, 5-th Berkeley Symposium on mathematics, 1, S. 281-298.
Painho, M. y Bação, F. (2000). *Using Genetic Algorithms in Clustering Problems*. Proceedings of the 5th International Conference on GeoComputation, University of Greenwich, UK.
Qin He, (1996). *A review of clustering algorithms as applied in IR*, UIUCLIS-1996/6+IGR, University of Illinois at Urbana-Champaign.
Rasmussen, E. (1992). *Clustering Algorithms*, W. B. Information Retrieval, Páginas 419-442. Prentice Hall, Eaglewood Cliffs, N. J.
Rüger, S. M. R. y Gauch, S. E. (2000). *Feature reduction for document clustering and classification*. Technical report, Imperial College, London, UK.
Schütze, Hinrich y Silverstein Craig (1997) *Projections for Efficient Document Clustering*, in Proceedings of ACM/SIGIR'97, pp.74-81.
Shannon, C.E. (1948) *A mathematical theory of communication*, Bell System Technical Journal, vol. 27, pp. 379-423 and 623-656, July and October.
Steinbach, M., Karypis, G., y Kumar, V. (2000). *A comparison of Document Clustering Techniques*. Technical Report #00-034. University of Minnesota.
Strehl, A., Ghosh, J. y Mooney, R. (2000). *Impact of Similarity Measures on Web-page Clustering*. AAAI-2000: Workshop of AI for Web Search.
Van Rijsbergen, C. J. (1979). *Information Retrieval*, Butterworths, London, 2da edición.
Wilf, H.S. (1986). *Algorithms and Complexity*, Prentice Hall.
Willet, P. (1998). *Recent trends in hierarchical document clustering: a critical review*. Information Processing and Management. 24:577-97.
Yang, Y. y Liu, Xin, (1999). *A re-examination of text categorization methods*. 22nd Annual SIGIR.
Zamir, Oren y Etzioni, Oren. (1998). *Web Document Clustering: A feasibility demonstration*. Proceedings of ACM/SIGIR'98.
Zamir, Oren y Etzioni, Oren. (1999). *Grouper: A Dynamic Clustering Interface to Web Search Results*. Proceedings of the Eighth International World Wide Web Conference.
Zhao, Y. y Karypis, G., (2001). *Criterion Functions for Document Clustering*. Technical Report #01-40, University of Minnesota.